CS 151 Complexity Theory **Spring 2023**

Problem Set 7

Out: May 25 *Due:* **June 1**

Reminder: you are encouraged to work in groups of two or three; however you must turn in your own write-up and note with whom you worked. You may consult the course notes and the optional text (Papadimitriou). The full honor code guidelines and collaboration policy can be found in the course syllabus.

Please attempt all problems. **Please turn in your solutions via Gradescope, by 1pm on the due date.**

1. Linearity Testing. Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a function, and suppose that

$$\Pr_{x,y}[f(x) + f(y) = f(x+y)] \geq 1 - \delta, \tag{7.1}$$

where x, y are chosen uniformly in $\mathbb{F}_2^n$. Define $\tilde{f} : \mathbb{F}_2^n \to \mathbb{F}_2$ by

$$\tilde{f}(x) = \text{majority}_y(f(x+y) - f(y)),$$

where we set $\tilde{f}(x) = 0$ if there is a tie.

(a) Prove that for all $x \in \mathbb{F}_2^n$, we have

$$\Pr_y[f(x+y) - f(y) = \tilde{f}(x)] \geq 1 - 4\delta.$$

Hint: start by relating the probability that the a random voter disagrees with the majority to the probability that two random voters disagree.

(b) Show that $\Pr_x[f(x) \neq \tilde{f}(x)] \leq 2\delta$. Hint: use Eq. (7.1) and the definition of $\tilde{f}$.

(c) Assume that $\delta < 1/14$. Prove that $\tilde{f}$ is linear; i.e., for all x, y we have

$$\tilde{f}(x) + \tilde{f}(y) = \tilde{f}(x+y).$$

Hint: Eq. (7.1) gives us

$$\begin{aligned} \Pr_{w,z}[f(w) + f(z) = f(w+z)] &\geq 1 - \delta \\ \Pr_{w,z}[f(x+w) + f(y+z) = f(x+y+w+z)] &\geq 1 - \delta. \end{aligned}$$

Now use part (a) to obtain statements about $\tilde{f}(x)$, $\tilde{f}(y)$ and $\tilde{f}(x+y)$.

(d) Given a function $f : \mathbb{F}_2^n \to \mathbb{F}$, the BLR linearity test on f is the following procedure: pick $x, y \in \mathbb{F}_2^n$ randomly; f passes the test if $f(x) + f(y) = f(x+y)$. Prove the following two statements:

Completeness: If f is linear, then f passes the test with probability 1.

Soundness: If f passes the test with probability $1-\delta$, then there exists a *linear* function $\tilde{f}$ satisfying $\Pr_x[f(x) = \tilde{f}(x)] \geq 1 - O(\delta)$.

2. Recall that a *clique* in an undirected graph $G = (V, E)$ is a subset $V' \subseteq V$ with edges between every pair of vertices in V'. We know that the language

$$\text{CLIQUE} = \{(G, k) : G \text{ has a clique of size } k\}$$

is **NP**-complete. You will show that there is some constant $\delta > 0$ for which CLIQUE is **NP**-hard to approximate to within N^δ in the following sense: if there is an N^δ-approximation algorithm for CLIQUE, then **NP** = **ZPP**. Here N is the length of the input (G, k).

The PCP Theorem implies that there is some constant $\epsilon > 0$ for which given a 3-CNF formula ϕ it is **NP**-hard to distinguish between the following two cases:

$$\begin{aligned} \text{YES} &: \ \phi \text{ is satisfiable} \\ \text{NO} &: \ \text{every assignment to } \phi \text{ satisfies at most a } (1-\epsilon) \text{ fraction of the clauses} \end{aligned}$$

Below you will describe a *randomized* transformation from an instance ϕ into a graph G whose intended effect is that a YES instance produces a graph with a large clique, while a NO instance produces a graph with only a very small clique. Here n is the number of variables in ϕ.

(a) Suppose ϕ is a NO instance, and consider the following probabilistic experiment: pick $\log_2 n$ clauses from ϕ uniformly at random, take their conjunction, and call this CNF ϕ_1; repeat n^3 times to get CNFs $\phi_1, \phi_2, \ldots, \phi_{n^3}$. Show that for a fixed assignment A:

$$\Pr[A \text{ satisfies at least } n^{3-\epsilon} \text{ of the } \phi_i] < e^{-n^2}.$$

Hint: What is the probability that A satisfies a given ϕ_i? What is the expected number of ϕ_i satisfied by A? You may want to use the fact that $(1-\epsilon)^{1/\epsilon} \leq 1/e$ for $1 > \epsilon > 0$, and the Chernoff bound: if X is the sum of independent 0/1 random variables with expected value $\mathrm{E}[X] \leq \mu$, then $\Pr[X > 2\mu] \leq e^{-\mu/3}$.

(b) Argue that the above randomized procedure produces from ϕ a collection of 3-CNFs $\phi_1, \phi_2, \ldots, \phi_{n^3}$ for which

i. ϕ is a YES instance $\Rightarrow$ $\Pr[\exists$ assignment A simultaneously satisfying all of the $\phi_i] = 1$, and

ii. ϕ is a NO instance $\Rightarrow$ $\Pr[$no assignment satisfies more than $n^{3-\epsilon}$ of the $\phi_i] \geq 1/2$.

(c) Describe an efficient deterministic procedure to construct a graph G from the collection of 3-CNFs in part (b) for which

i. $\exists$ assignment A simultaneously satisfying all of the $\phi_i \Rightarrow G$ has a clique of size n^3, and

ii. no assignment satisfies more than $n^{3-\epsilon}$ of the $\phi_i \Rightarrow$ no clique in G has size greater than $n^{3-\epsilon}$.

(d) Prove that there exists a constant $\delta > 0$ for which an N^δ-approximation algorithm for CLIQUE implies that $\mathbf{NP} = \mathbf{ZPP}$, where N is the length of the input.

3. **Optional for extra credit**. This problem concerns the circuit class **ACC**: polynomial-sized, constant-depth circuits with unbounded fan-in AND and OR gates, NOT gates, and unbounded fan-in MOD-m gates, which output 1 iff the number of true inputs to the gate is divisible by m. This appears to be a very weak class (e.g., without the MOD-m gates, it is known that it incapable of even computing the parity function). However, until late 2010, no one knew how to prove that **NEXP** is not in **ACC**! This problem follows Ryan William's breakthrough result that finally shows that **NEXP** is not in **ACC**. You will prove the slightly easier statement that $\mathbf{E}^{\mathbf{NP}}$ is not in **ACC**, which contains all of the main ideas. Recall that $\mathbf{E} = \cup_{k\geq 1}\mathbf{TIME}(2^{kn})$.

   (a) We say "**ACC**-type circuit" to mean a constant-depth circuit with the same allowed gates as for **ACC**, but not necessarily polynomial-size. We use quasipoly(n) as shorthand for $2^{\log^{O(1)} n}$. Using the following theorem, show how to obtain a list of *all* evaluations of an **ACC**-type circuit C with n inputs in time $O(2^n \cdot \text{poly}(n, \log|C|) + \text{quasipoly}(|C|))$. Note this is much better than the $O(2^n \cdot |C|)$ running time of the obvious algorithm when $|C|$ is large.

   Theorem 7.1 *Given an* **ACC**-*type circuit* C, *there is a deterministic procedure running in time* $O(\text{quasipoly}(|C|) + 2^n \cdot \text{poly}(n))$ *that produces from* C *a multilinear polynomial* f *with nonnegative coefficients (presented as a vector of* 2^n *coefficients) and a poly-time computable function* $T : \{0, 1, \ldots, \ell\} \to \{0, 1\}$, *with* $\ell \leq \text{quasipoly}(|C|)$, *such that* $C(x) = T(f(x))$. *Here we are identifying Boolean values FALSE and TRUE with* $0, 1 \in \mathbb{Z}$.

   Hint: First compute $f(x)$ for all $x \in \{0,1\}^n$ with $O(n2^n)$ arithmetic operations using a divide and conquer algorithm.

   (b) Give a deterministic algorithm that is given an **ACC** circuit C with n inputs and determines if it is satisfiable in time $O(2^{n-n^\delta})$ for some constant $\delta > 0$. Hint: plug in all possible values for the first $n' < n$ variables to form new **ACC**-type circuit.

   (c) We know from Lecture 2 that SUCCINCT 3-SAT is **NEXP**-complete. A more careful reduction gives the following: for every language $L \in \mathbf{NTIME}(2^n)$, there is a reduction from L to SUCCINCT 3-SAT mapping instances x of length n to circuits C_x of polynomial size, with $m = n + 5\log n$ inputs. Such a circuit succinctly encodes a 3-SAT formula ϕ_x of length 2^m.

   Show that if $\mathbf{E}^{\mathbf{NP}} \subseteq \mathbf{ACC}$, then SUCCINCT 3-SAT has *succinct witnesses*: for every $x \in L$, there is an **ACC** circuit W_x such that assigning the i-th variable in ϕ_x the value $W_x(i)$ satisfies ϕ_x.

   (d) Show that if $\mathbf{E}^{\mathbf{NP}} \subseteq \mathbf{ACC}$, then every language $L \in \mathbf{NTIME}(2^n)$ can be simulated in $\mathbf{NTIME}(2^{n-n^\delta})$ for some constant $\delta > 0$, contradicting the **NTIME** hierarchy.

   Hint: given the circuit C_x from the previous part, guess **ACC** circuits D (intended to be equivalent to C_x), G (intended to encode C_x gate by gate – i.e., given the index of a gate, it output the type of gate, and the indices of its at most 2 inputs), V (intended to encode the gate-by-gate evaluation of C_x – i.e., given an input z and the index of a gate, it output that gate's value when C_x is evaluated on z) and W_x (intended to be a succinct witness for ϕ_x). Use part (b) twice, once to verify that D is equivalent to C_x (using G and V to express this equivalence using only **ACC** circuits) and once to verify that W_x encodes a satisfying assignment to ϕ_x.